Implementing RNN with Non-Randomized GA for the Storage of Static Image Patterns

Raj Kumar Goel[1]*, Ganesh Kumar Dixit[2], Saurabh Shrivastava[3], Manu Pratap Singh[4], and Shweta Vishnoi[5]

[1,5]Noida Institute of Engineering & Technology, Greater Noida, India
[2]B.S.A. College, Mathura (UP), India
[3]Bundelkhand University, Jhansi (U.P), India
[4]Dr. B.R. Ambedkar University, Agra, UP, India
*raj9921@yahoo.com

Abstract: The hybridization of evolutionary technology has been extensively used to enhance the performance of recurrent type neural networks (RTNN) for storing patterns and their recalling. Several experiments have been done to link evolutionary processes such as genetic algorithm (GA) with RTNN regarding the connection weight among the processing elements. This integration strengthens the efficiency of the Recurrent neural network (RNN) to effectively recall the increased capacity and patterns of sample storage to reduce the flaw of local minima. Bipolar product rule (BPR) has been applied predominantly for pattern storage, and GA are further used to develop the weight matrix to explore the global optimal solution reflecting the correct invocation of the storage pattern. Here, Edge Detection (ED) and self-organizing map (SOM) methods are applied for the purpose of feature extraction. The modified BPR and GA have been employed to store patterns, and recalling respectively. The proposed hybrid RTNN performance is examined for the handwritten Greek symbols.

Keywords: Hybrid Evolutionary Techniques, RNN topology, Non-random GA, Pattern Storage Network.

1. Introduction

An ANN with the recurrent topology is referred as recurrent neural network. It is analogous to a feed-forward network with no restrictions on feedback loops. The transmission of information is both forward and backward. This makes an inner state of the network that permits it to show dynamic temporal behavior. Recurrent networks may utilize their internal memory for processing any sequence of inputs [1,2].

RNN with symmetrical weight, bipolar processing unit and asynchronous activation dynamics with stochastic or deterministic update is extensively used for storing and retrieving patterns. RNN with bipolar threshold function and symmetrical weights shows the characteristics of associative memory [3,4]. It has been found that the use of Hebbian learning rules (HLR) to encode pattern information in RNN will cause false minima problems with a large network. The recall process combined with GA can minimize the problem of false minima. This problem can be reduced with the amalgamation of GA for retrieving process [5].

An NN can easily handle inaccurate, probabilistic, noisy and fuzzy information. The NN is a distributed information system in which information is stored throughout the network, distributed over the network structure. The NN, namely the Hopfield network, can be applied as associative information storage. At one point in time, the N neuron network state will be expressed by an N vector with binary components, which can be considered as a binary code of few information. As part of its free mobility, the network evolves from the initial phase to one of the certain points. Hereby, it assigns an initial phase to encode garbled, incomplete or distorted information to a fixed point where it can encode complete, complex-free information. So that, this allows information to be "recovered" from noisy or partially imperfect versions [6].

Received: December 29th, 2018. Accepted: December 9th, 2020
DOI: 10.15676/ijeei.2020.12.4.16

Genetic algorithm has two Common applications such as a function optimizer and evolving organisms which work well in a particular environment. In any case, GA is based on Darwinian theory of evolution, the most relevant (natural selection) theory for survival. The neural network, on the other hand, the itself seems useful in the form of a mechanism of control of organisms (for example, one should avoid the threat of organism and seek food). Here, population-generating method, crossover operator and fitness assessment function are explored for generating the optimal weight matrix to apply GA [7,8].

Neural network stores the patterns using Hebbian learning to encode the pattern information in the connection strength or weight matrix. The encoded weight matrix corresponds to the sample pattern or training set exhibits the less storage capacity, i.e. 1.5N where N is the number of units in the network. Pseudo inverse rule has been proposed to enhance the storage capacity, recalling efficiency and pattern correction [9]. It exhibits 0.5N storage capacity of the network but, on the other hand, the recalling efficiency of network decreases, as the storage capacity increases [10].

2. Related Work

The efficiency of pattern recall can be improved by integrating GA with RNN. The amalgamation of GA and NN is done for pattern recalling, while HLR is used for pattern storage, such type of hybridization shows better performance compared to conventional neural feedback networks [11]. The major problem in the method mentioned above is that the capacity increases for noiseless patterns only but it fails to obtain the satisfactory results for noisy patterns [12].

This study is an attempt to improve storage competence and efficiency reminiscent of both the noisy and noiseless patterns by introducing SOM as a feature extraction technique with modified HLR for pattern storage and the inclusion of GA for pattern retrieval. In order to effectively reminiscent the input prototype pattern, the optimal weight matrix can be obtained by applying GA and RNN. Scanned images of Greek symbols have been used as an input stimuli. Here, the GA starts not from the random population of the weight matrix, but from the encoded weight matrix of pattern information. We investigate GA with a suboptimal weight matrix and as well as mutation, local and global crossover operator are used to generate the weight matrices population. Each population is tested by the fitness evaluation function. The weight matrices (selected population) also produce larger weight matrices populations. Correct recalled samples or global optimal minima are obtained by successive repetition of above-mentioned process. It has been seen that the efficiency of pattern storage networks also modified by the feature extraction technique. In this area, a lot of effort is focused on the use of various feature extraction techniques to minimize the possibility of errors in pattern recall [13-16].

GA is used to evolve the population of encoded weight vectors which are generated with recurrent neural network. Iterative training algorithm can be applied to recurrent network of various sizes and tested for three different sizes. Information obtained is used to find network model critical dynamic properties. The integration of GA in neural network has already been intensively discussed for the pattern recalling process [17,18]. As compared to hybrid evolutionary algorithms and GA, it has been observed that back propagation algorithms require more execution time and space. GA has also been used to improve the network capacity and recalling efficiency [19].

Section 3 elaborates feature extraction and RNN topology. implementation of RNN with GA will be described in Section 4. Results of experiments with discussion are provided in section 5. Section 6 summarizes the work with future scope followed by references.

3. Proposed Method

Figure 1 shows the scanned Greek symbol images used in this experiment. At first, gray images are acquired by converting these scanned RGB static images, and then converting the obtained grayscale images into binary images with a size of 30×30. After binarization of image, we apply the ED method to show the defined base of the image as depicted in figure 2. ED consists a process of distinguishing, extracting, and highlighting useful image features [20]. For

representing pattern vector these images are converted into 900×1 matrix. Finally, there are 24 sample images of Greek alphabet symbols, so the size of the training set is 900 × 24. Also, this training set is provided to the SOM to extract information about the feature map coding mode. SOM creates and organizes description of data with linear vector quantization, i.e. it presents the input pattern of any dimensions into 1D or 2D array of processing units. Figure 3 shows the steps of the proposed research study using the block diagram.

Learning process of SOM considers the weight update for the winner unit and also the weights of neighborhood interconnected units as:

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)\left(x_i - w_{ij}(t)\right) \forall i \in h_{ij}(t) \quad (1)$$

Among these , $\eta(t)$ is the learning rate, x_i is the input pattern and $h_{ij}(t)$ is the neighborhood function for the area around the winner processes unit.

In this experiment, the SOM considered the size of 900 × 24 input pattern vector and was adjusted to create a local response to the presented input pattern vector, thus reflecting the topological order of the input vector. This sequencing of input pattern vectors denotes the feature location of the training set. Hereby, the feature vector extracted from the 10 × 10 dimensional SOM describes the feature location of the 900 × 24-dimensional training set.

Features extracted from SOM in the form of 100 × 24 code words are now converted into bipolar state and presented to the recurrent neural network for storage.

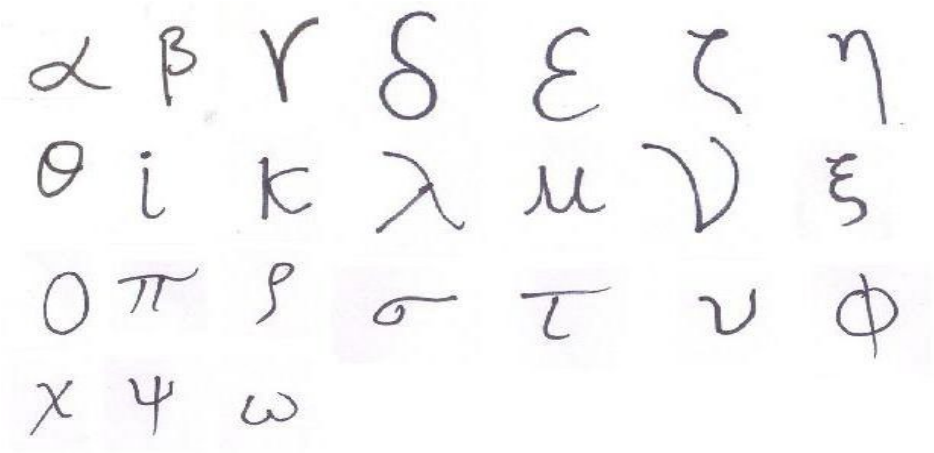


Figure 1. Handwritten scanned Greek symbol images

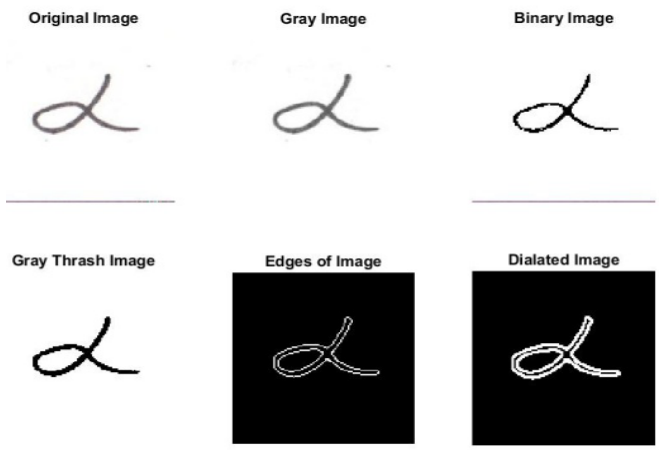


Figure 2. Several phases of images used in feature processing.

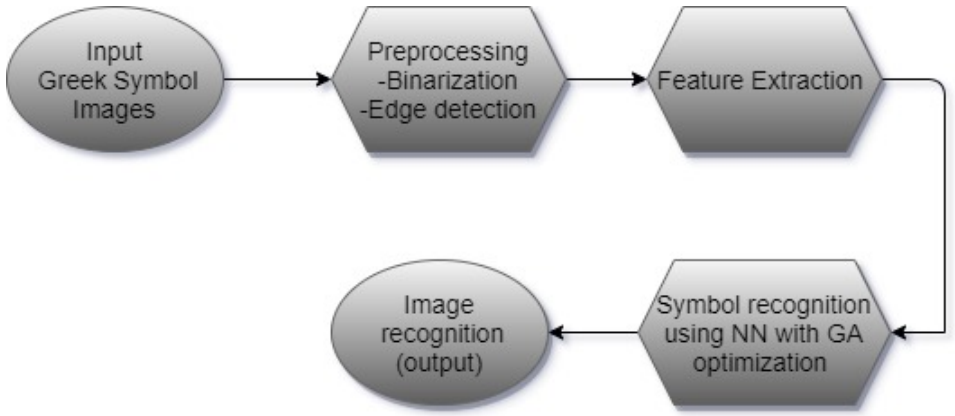


Figure 3. Steps of the proposed research study

A. Recurrent Neural Network

There are N (100=10×10) units in the given RNN model. By using modified HLR, each of N×1 (100×1) Input pattern is provided to the network for storage. Now, consider the stored L patterns, which have been encoded in an N×N weight matrix using a modified HLR.

$$W = \frac{1}{N}[PP^{T} + PP^{\dagger}] \quad (2)$$

Here P denotes the size of N×L input pattern vector, N is the number of units in the network, and P^{T} is the transpose of the P and $P^{\dagger}$denotes the pseudo inverse of P.

The pattern set P can be defined as

$$P = \begin{bmatrix} a_1^1 & a_1^2 & a_1^3 & \dots & a_1^L \\ a_2^1 & a_2^2 & a_2^3 & \dots & a_N^L \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ a_N^1 & a_N^2 & a_N^3 & . & a_N^L \end{bmatrix}_{N\times L} \quad (3)$$

Where $a_1, a_2 \dots a_n$ are the pattern vectors provided to the network. Therefore, after the encoding of pattern set using equation (2), the recalling process corresponds to input noisy pattern starts. The units of NN have been initialized with input pattern as:

$$S_i(0) = a^{L}{}_{i+\epsilon_i} \forall i = 1 \text{ to } N \quad (4)$$

Herein, ϵ_i represents the noise in the i^{th} element of L^{th} pattern.

The activation dynamics of RNN move the network to the stable state i.e.

$$S_i(t+1) = S_i(t) \quad (5)$$

or

$$\text{Sgn}\left(\sum_{j=1}^{N} w_{ij}(a_i^L + \epsilon_i)\right) = a_i^L \forall i, j = 1,2 \dots . . N \quad (6)$$

RNN explores the stable condition which reflects the recalled pattern.

Most of the time for larger N, network exhibits the problem of spurious minima or false minima. The problem of false minima limits the performance of RNN for recalling. Therefore, to improve the efficiency of NN for pattern recalling skipping, the false minima of GA has been incorporated with neural network.

Figure 4 shows the RNN topology, in which every pair of distinct artificial neurons in a recurrent network has a connecting synapse. An RNN is a type of ANN whose artificial synapses form an oriented graph along a sequence. This function makes it a real dynamic system. RNNs use their internal state to process data sequences. This network is entirely interconnected and determining the weight matrix is the most important tasks to use for any application. This property makes them suitable for connected handwriting recognition [21, 22].

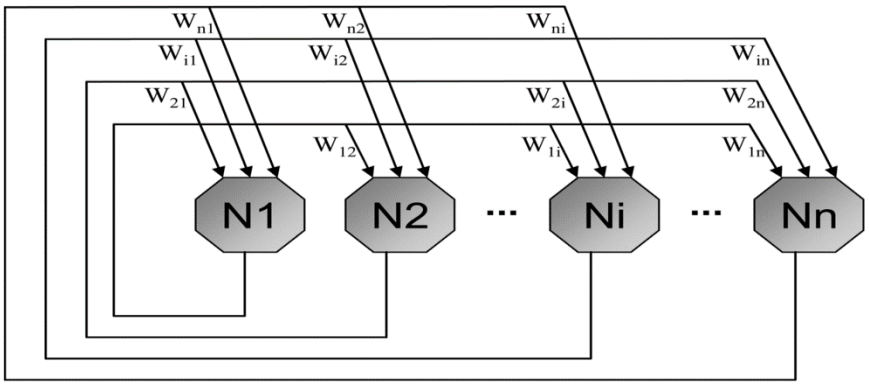


Figure 4. A Recurrent network made of n neurons.

The parameters used in the experiments are mentioned in table 1 and table 2

Table 1. Parameters used for Hebbian learning rule (RNN)

Parameter	Value
Initializing state (neurons)	Randomly generated values (-1 Or 1)
Threshold values (neurons)	0.00

Table 2: Parameters used for GA

Parameter	Value
Initializing state (neurons)	Randomly generated values (-1 Or 1)
Threshold values (neurons)	0.00
Mutation population size	N+1
Mutation probability	0.5
Crossover population size	N*N

4. Implementation

In simulation design, the RNN architecture is implemented by using 100 bipolar processing units, asynchronous state change, symmetric weights on interconnections among processing units, deterministic updating and modified bipolar out product learning rule for storing the pattern information of scanned static images of Greek symbol. The static Greek images are preprocessed and filtered through ED and SOM to build the pattern information for storage. In the training set, by using modified bipolar out product learning rule, equation 2 outlines the pattern information, which is encoded as the connection strength among the processing unit of the network

Therefore, a parent weight matrix was created for the pattern vectors of training set. GA used this parent weight matrix as an initial population or solution. Hence, the GA was used here to retrieve the pattern vector for the prototype input pattern vector presented. It begins with the

suboptimal parent weight matrix and uses three GA operators namely mutation, crossover, and fitness evaluation to evolve the weight matrix.

A. Mutation

This operator is initially applied to the encoded parent weight matrix of N×N (100×100) order with Pm <0.05. It generates the same weight matrix of order M, namely N×N. It chooses the non-zero individuals (allele)from the weight matrix chromosome for random modification. The modification of the selected alleles is implemented by several arithmetic operators (addition, subtraction and multiplication). Therefore, the change in the randomly selected allele in the current approach is not random, but rather it is modified with mathematical criteria. The process for using the mutation operator can be considered as follows:

For i=0 to N
For j=0 to N
If $(w_{ij}^{old} \neq 0)$ then
$w_{ij}^{new} = (\sum_{i,j=0}^{N} w_{ij}^{old})$ ⒽR^{VM}
Else

$w_{ij}^{new} = w_{ij}^{old}$

End if

Among these, Ⓗ is the applying mathematical operator and R^{VM} denotes the method of modification. So that, we obtain M weight matrices populations from the parent weight matrix of the equation. Equation 4 after applying the mutation operators is represented thus:

$$W = \{w_{MXN}^{1}, w_{MXN}^{2}, w_{MXN}^{3} \dots \dots w_{MXN}^{M}\} \quad (7)$$

B. Crossover

In this methodology, all M population matrices are divided into quadrants of fixed size S×S. Each quadrant matrix contains chromosomes. Crossover can be applied in two steps, namely local and the global crossover. In local crossover, the uniform crossover would be implemented by exchanging one quadrant for another, producing a new population of weight matrix of N×N size. In the global crossover, we randomly choose two populations of weight matrices from the M population. In these two selected weight matrices, we again randomly choose a quadrant of S×S order from a matrix and swap it with another randomly picked quadrant from another matrix of the same size to generate a new population of N×N size weight matrix as

$$w_{N\times N}^{i} = \begin{bmatrix} w_{11}^{i} & w_{12}^{i} & \dots \dots w_{1N}^{i} \\ w_{21}^{i} & w_{22}^{i} & \dots \dots w_{2N}^{i} \\ \dots & & \dots \\ w_{N,1}^{i} & w_{N,2}^{i} & \dots . w_{N,N}^{i} \end{bmatrix}_{N\times N}$$ Where i= 1, 2, 3 ….N

Local crossover:

$w_{1,N\times N}^{new} = w_{s\times s}^{R1} \leftrightarrow w_{s\times s}^{R2}$ Herein, R1, R2 are any randomly chosen quadrant from same weight matrix. Likewise,

$w_{2,N\times N}^{new} = w_{s\times s}^{R1} \leftrightarrow w_{s\times s}^{R2}$

……

$w_{N,N\times N}^{new} = w_{s\times s}^{R1} \leftrightarrow w_{s\times s}^{R2}$

Hence

$$w^{newL} = \{w_{1,N\times N}^{newL}, w_{2,N\times N}^{newL}, w_{3,N\times N}^{newL} \dots\dots w_{M,N\times N}^{new}\} \quad (8)$$

Here $w_{1,N\times N}^{newL}$, $w_{2,N\times N}^{newL}$ … … $w_{M,N\times N}^{newL}$ are recently created weight matrices after local crossover, $w_{s\times s}^{R1}$ and $w_{s\times s}^{R2}$ are two randomly selected sub-weight matrices from the old mutated population to accomplish local crossover operation for exchange.

b. Global crossover:

$w_{1,N\times N}^{new} = w^{i}{}_{s\times s}^{R_1R_2} \leftarrow w^{j}{}_{s\times s}^{R_1R_2}$ For i ≠ j and i , j= 1, 2, 3, … …M

$w_{2,N\times N}^{new} = w^{i}{}_{s\times s}^{R_1R_2} \leftrightarrow w^{j}{}_{s\times s}^{R_1R_2}$

:

:

$w_{n,N\times N}^{new} = w^{i}{}_{s\times s}^{R_1R_2} \leftrightarrow w^{j}{}_{s\times s}^{R_1R_2}$

Where $w^{i}{}_{s\times s}^{R_1R_2}$, $w^{j}{}_{s\times s}^{R_1R_2}$ two are any randomly picked quadrant from dissimilar weight matrices. So, after applying this global crossover, we get new weight matrices as

$$w^{newG} = \left\{w_{1,N\times N}^{newG}, w_{2,N\times N}^{newG} \dots \dots \dots \dots \dots w_{m,N\times N}^{newG}\right\} \quad (9)$$

Hence, we apply the crossover operations m times as:

$$w^{C} = \bigcup_{n=1}^{N} w_n \bigcup_{k=1}^{m} w_{k,N\times N}^{newL} \bigcup_{t=1}^{P} w_{t,N\times N}^{newG} \quad (10)$$

Where w_n are mutations after N+1 time and

$$w_{t,N\times N}^{newG} = ®_i(w_{s\times s}^{t}) \Leftrightarrow ®_j(w_{s\times s}^{t}) \quad (11)$$

Among them, i ≠ j and $®_i$, $®_j$ are randomly chosen matrices of order S×S

$$w_{K,N\times N}^{newL} = w_{S\times S}^{K,R_j} \leftrightarrow w_{S\times S}^{K,R_i} \quad (12)$$

Where $®_i$, and $®_j$ are the selected S×S matrices from different chromosomes.

Thus, we obtain the population of new weight matrices after this crossover operations as specified in equation (11) and (12).

C. Fitness Function

For each weight matrix, as given in Equation 3, a fitness function (f) has been built for the entire pattern sample P. Assign each weight matrix to the network of a given pattern set or prototype pattern set (which contains the noisy sample pattern), in this way, the recalled pattern vector is obtained On the basis of regression value between the expected and the actual pattern, the performance of the retrieval network is measured.

Regression value (R) for each weight matrix is characterized and further obtain the mean of these values as:

$$f_1(w) = \text{Avg}[Rw_1^c, Rw_2^c, \dots \dots \dots . Rw_m^c] \quad (13)$$

Where w_1^c , w_2^c , … . . w_m^c the populations of weight matrices attained from equation 11.

The fitness evaluation function f(w) for each weight matrix is as follows:

$$f(w) = \begin{cases} 1 \text{ if} R_{w_i}^c > f1(w) \forall i = 1 \text{ to m} . \\ 0 \text{ if} R_{w_i}^c \leq f1(w) \forall i = 1 \text{ to m}. \end{cases}$$

Therefore, we select those weight matrix populations whose NN performance is 1 in the recall process, i.e. f(w)=1. Only the selected weight matrix is implicitly used in the next iteration or cycle of GA to generate a new weight matrix filling.

5. Experiment and Results

Simulation result presented here exhibits that the performance of hybrid approach, i.e. the combination of GA and RNN for recalling process improves the performance of RNN significantly. The simulated output produces the regression value (R=1) for all the pattern vector of training set. The performance deteriorates for the noisy or erroneous pattern. In the proposed framework of experiment, the presented prototype input pattern consists of 10%, to 50% of noise or distortion with the step size of 10%. Figure 5 shows the results of recalling in regression value

for the pattern which contains 10% error with modified Hebbian rule and sub optimal GA. In the same way, from figure 6 to 9 show the recalling for the presented patterns of 20% to 50% errors with step size of 10%. Table 3 shows RNN performance evaluation to recall using SOM and GA technique. From the result it can be seen that the modified HLR gives better response for a noisy pattern in most of the cases. As we can see, for 30% error, improvement is approximately 2% while for 10%, 20% and 50%, it is hardly near to 1%. In case of 40% error, there is not a significant difference.

Table 3. RNN performance evaluation to recall using SOM and GA technique.

Comparison	On 10% Error	On 20% Error	On 30% Error	On 40% Error	On 50% Error
Modified HLR without GA	0.96880	0.87580	0.69450	0.46250	0.14320
Modified HLR with GA	0.97633	0.88475	0.71874	0.46632	0.15293
variation	0.00753	0.00895	0.02424	0.00382	0.00973

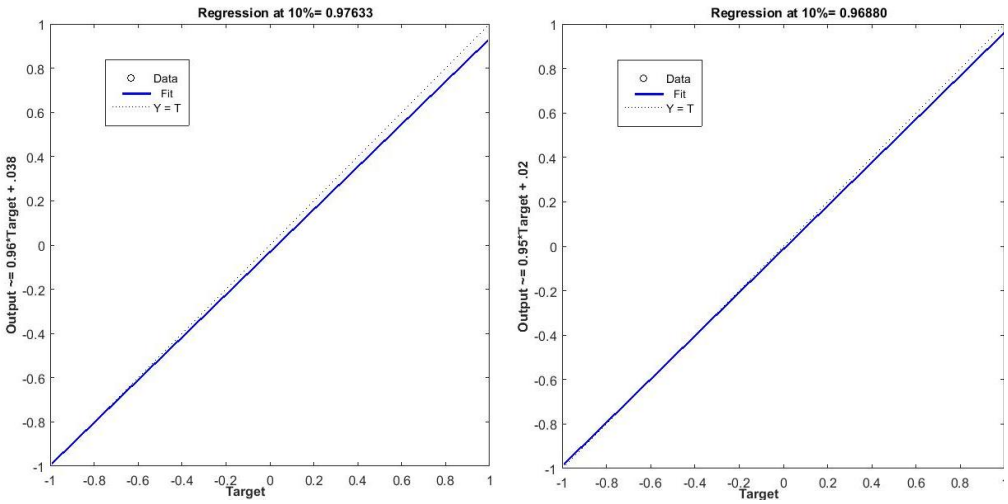


Figure 5. Comparative analysis of recalling by using modified HLR with and without GA at 10% Error.

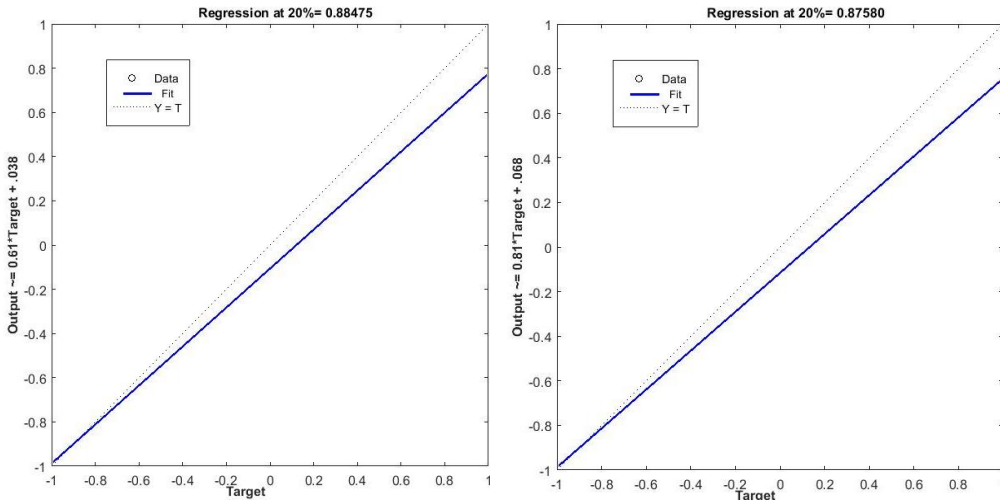


Figure 6. Comparative analysis of recalling by using modified HLR with and without GA at 20% Error

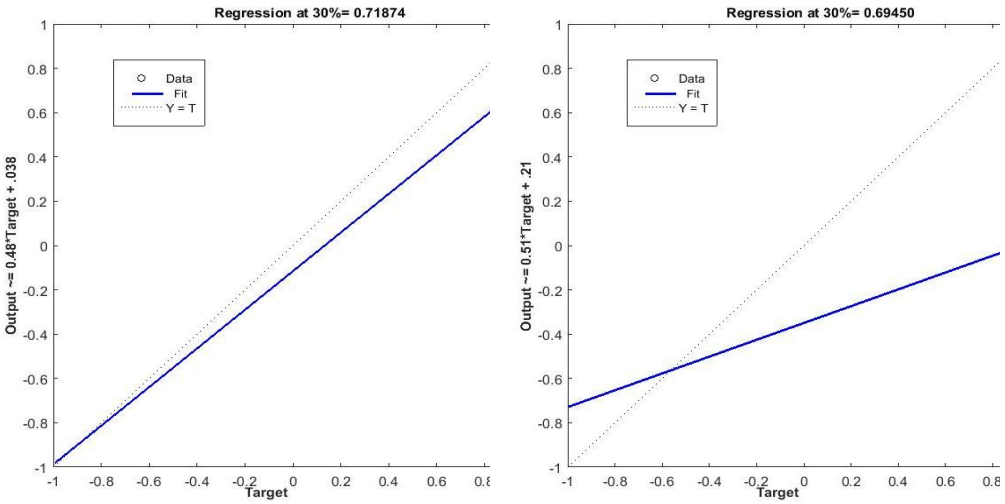


Figure 7. Comparative analysis of recalling by using modified HLR with and without GA at 30% Error

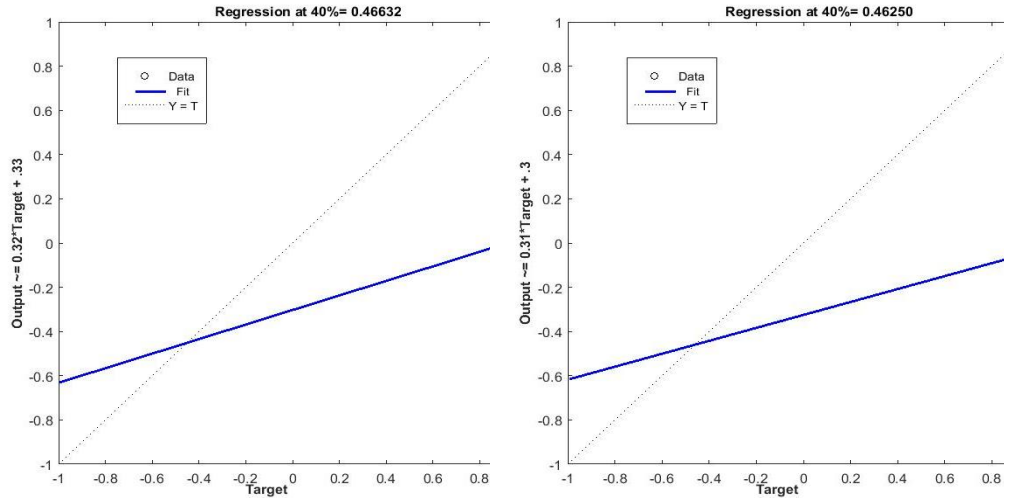


Figure 8. Comparative analysis of recalling by using modified HLR with and without GA at 40% Error

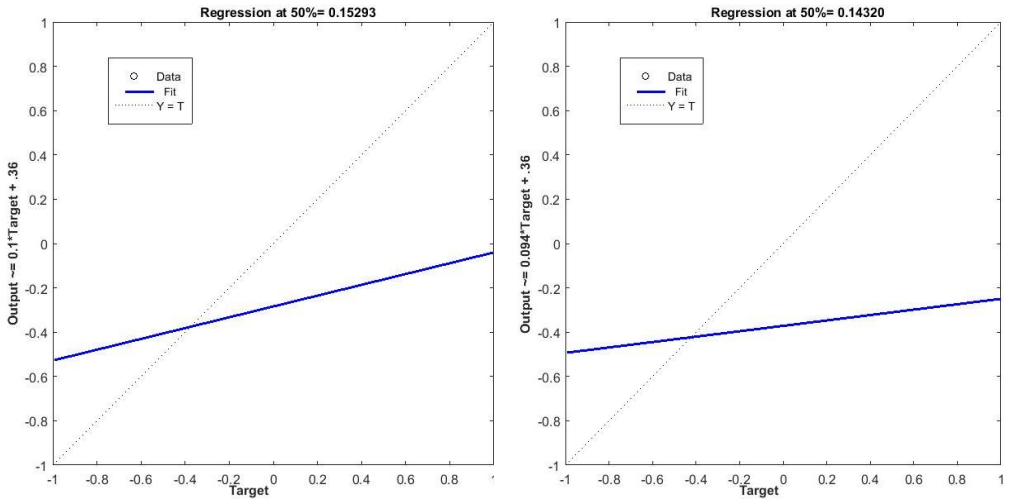


Figure 9. Comparative analysis of recalling by using modified HLR with and without GA at 50% Error.

6. Discussion and Evaluation

As shown in Figure 1, simulation results of 24 static Greek symbol images have been obtained. These images have been stored in the RTNN using modified HLR. The network recalling performance has been assessed for noise pattern vectors. GA has been employed with RNN for recalling purpose. Here, 20 iteration of GA is run and the size of population is fixed to 100. The obtained results exhibit the superior performance of the NN with GA to retrieve the correct pattern even if the prototype input pattern presented contained noise or errors. It has been observed that the method proposed by the Singh et al. [13] and Sarangi et al [23], has shown that, recall efficiency of RNN effective up to 40% of distorted pattern. By applying non-randomize GAs to optimize the performance of RNN, it can do well both the noiseless and up to 50% of noisy patterns. From the result, it has been seen that GA with NN is capable of remembering the correct output pattern up to 50% of noise in the input pattern. Figure 10 shows comparative analysis of recalling by using modified HLR with and without GA of Greek symbol images.

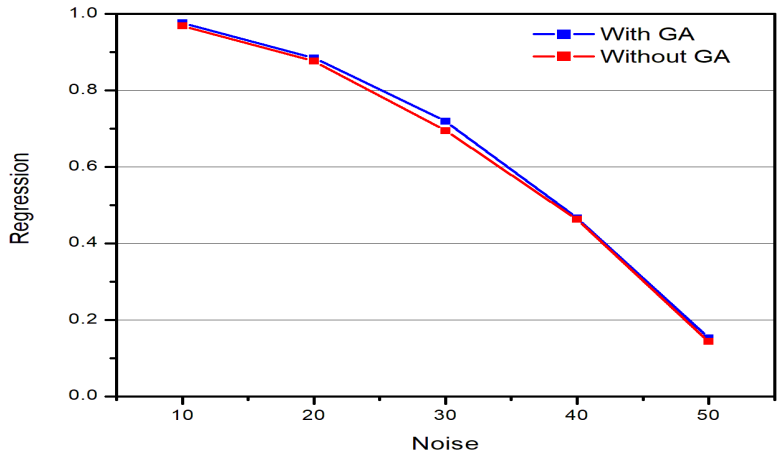


Figure 10. Comparative analysis of recalling by using modified HLR with and without GA of Greek symbol images.

7. Conclusion and future scope

The considered network is very large and the numbers of patterns are very low, so the network traps in false minima during recalling process for presented prototype input patterns. The problem of false minima has been minimized with the integration of GA for recalling purpose. From simulation results it has been stated that the pattern stored with modified bipolar product rule works superior for recalling of presented input prototype patterns with noise from 10% to 50% with step size of 10% if the GA has been integrated in the recall process. Result shows that the effectiveness of the hybridization make better if global crossover govern the local crossover. The proposed genetic algorithm starts from the weight matrix which has been constructed by the modified bipolar product law rather than the random population. Thus, the proposed genetic algorithm has been considered as non-random because it starts to explore the weight space from encoded weight matrix.

Thus, using fitness evaluation function, GA randomness minimizes as the population is filtered efficiently. Therefore, the smaller populations will be created and the trained produced population is better suited and appropriate for the solution which results in reduction of searching time. Therefore, we achieve the reduced time in searching and randomness by implementing GA resulting in optimal solution.

RNN services can also use more advanced feature extraction methods to analyze a large number of images. By using hybridization evolutionary technique to fetch the patterns, the efficiency of RNN can be further improved. In future, this concept can be implemented for various pattern recognition problems in SOM–RNN to further improve the recalling efficiencies.

8. Acknowledgements

Research study was supported by UGC, India, under the grant UGC-Ref No-3485.

9. References

[1]. Noman, F., Alkawsi, G., Alkahtani, A.A., Al-Shetwi, A.Q., Tiong, S.K., Alalwan, N., Ekanayake, J. and Alzahrani, A.I., 2020. Multistep short-term wind speed prediction using nonlinear auto-regressive neural network with exogenous variable selection. Alexandria Engineering Journal.

[2]. R. J. Colvin, Modelling and analyzing neural networks using a hybrid process algebra. Theoretical Computer Science, Volume 1, pp 1–50., 2015.

[3]. N. Davey, S.P. Hunt and R. Adams, High capacity recurrent associative memories. Neurocomputing IJON, Volume 62, pp 459–491, 2004.

[4]. J.J. Hopfield, Neural networks: Neural networks physical systems with emergent collective computational abilities. Proceedings of the National Academy Sciences. USA, Volume 79, pp 2554–2558, 1982.
[5]. S. Kumar and M.P. Singh, Study of Hopfield neural network with sub-optimal and random GA for pattern recalling of English characters. Applied Soft Computing Journal, 12(8), pp 2593-2600, 2012.
[6]. Goulon-Sigwalt-Abram, Aurélie, Arthur Duprat, and Gérard Dreyfus. "From hopfield nets to recursive networks to graph machines: numerical machine learning for structured data." Theoretical computer science 344, no. 2-3 (2005): 298-334.
[7]. Khan, A., Qureshi, A.S., Wahab, N., Hussain, M. and Hamza, M.Y., 2019. A recent survey on the applications of genetic programming in image processing. arXiv preprint arXiv:1901.07387.
[8]. Ahlawat, S. and Rishi, R., 2019. A genetic algorithm-based feature selection for handwritten digit recognition. Recent Patents on Computer Science, 12(4), pp.304-316.
[9]. F.E. Streib, Active learning in recurrent neural networks facilitated by a Hebb-like learning rule with memory. Neural Information Processing. Letters and Reviews, Volume 9, Number 2, pp 31–40, 2005.
[10]. S.I. Amari, Learning patterns and pattern sequences by self-organizing nets of threshold elements. IEEE Transactions on Computing, Volume 21, pp 1197-1206, 1972.
[11]. A.M. Mangal, and M.P. Singh, Analysis of Pattern Classification for the Multidimensional Parity-Bit- Checking Problem with Hybrid Evolutionary Feed-Forward Neural Network. Neurocomputing, Volume70, pp 1511–24, 2007.
[12]. B.M Forrest, Content-addressability and learning in neural networks. Journal of Physics A: Mathematical and General, Volume 21, Number 1, pp 245-25, 1988.
[13]. M.P. Singh and R.S Dixit, Optimization of stochastic networks using simulated annealing for the storage and recalling of compressed images using SOM. Engineering Applications of Artificial Intelligence, Volume26, Number 10, pp 2383–2396, 2013.
[14]. S. Sasikala, S. Appavu and S. Geetha, A novel adaptive feature selector for supervised classification. Information Processing Letters, Volume 117, pp 25–34, 2017.
[15]. Saptawati, G.A.P., Set of Frequent Word Sequence (SFWS) as Document Model for Feature Based Document Clustering. International Journal on Electrical Engineering & Informatics, 11(4) pp. 822-832, 2019.
[16]. K. Hosseini, Short Circuit Fauiolt Classification and Location in Transmission Lines Using a Combination of Wavelet Transform and Support Vector Machines. International Journal on Electrical Engineering and Informatics, 7(2), pp 353-365, 2015.
[17]. Aini, Q., Affandi, A., Setijadi, E., Terasawa, T. and Purnomo, M.H., 2017. Optimization of Efficient Route and Profitable Fish Aggregating Device Based on Firefly and Genetic Algorithms in Maritime. International Journal on Electrical Engineering and Informatics, 9(4), pp.747-761.
[18]. J.S Huang and H.C. Liu, Object recognition using GAs with a Hopfield's neural model. Expert Systems with Applications, Volume13, Number 3, pp 191–199, 1997.
[19]. S. Shrivastava and M.P. Singh, Performance evaluation of feed-forward-neural-network with soft computing techniques for hand-written English alphabets. Applied Soft Computing,11(1), pp 1156–1182, 2011.
[20]. B. Jacob and H. Gao, Robust edge detector based on anisotropic diffusion-driven process. Information Processing Letters, Volume 1, pp 1–6, 2016.
[21]. Sharma, R. and Kaushik, B., 2020. Offline recognition of handwritten Indic scripts: A state-of-the-art survey and future perspectives. Computer Science Review, 38, p.100302.
[22]. Bugeja, M., Dingli, A. and Seychell, D., 2020. An Overview of Handwritten Character Recognition Systems for Historical Documents. In Rediscovering Heritage Through Technology (pp. 3-23). Springer, Cham.

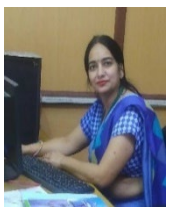
[23]. Sarangi, P.K., Sahoo, A.K. and Ahmed, P., Recognition of isolated handwritten oriya numerals using hopfield neural network. International Journal of Computer Applications, 40(8), pp.36-42. 2012.

Raj Kumar Goel is working as Associate Professor in the Department of CSE at NIET, Greater Noida. He has total teaching experience of sixteen years. He received his M.tech(CSE) from GBTU, Lucknow. He has completed his Ph.D in 2019 from Bundelkhand University, India. He has more than 22 research papers in journals of international and national repute. His research interests include neural network, software engineering, artificial intelligence and digital image processing.

Ganesh Kumar Dixit received his Bachelor degree in Science and PG Diploma in Computer Programming from Dr. B. R. A. University, Agra in 1998 and 1999 respectively, and MCA from IGNOU in 2005. He has completed his Ph.D from Mewar University in 2017. He started his academic career as Assistant Professor in Department of Computer Science, BSA (PG) College, Mathura (UP) in 2007. His research interest is in Soft Computing and Image Processing.

Saurabh Shrivastava received his MCA from MITS Gwalior in the year 1998. After serving one year in software industry he joined Bundelkhand University Jhansi as assistant professor in the year 1999. He received his doctorate degree in the year 2009 from Bundelkhand University Jhansi. His research areas are soft computing, data mining and wireless sensor network. Presently he is working as associate professor in Bundelkhand University Jhansi.

Manu Pratap Singh received his Ph.D. in Computer science from Kumaun University Nanital, Uttarakhand, India, in 2001. He has completed his Master of Science in Computer Science from Allahabad University, Allahabad in 1995. Further he obtained the M. Tech. in Information technology from Mysore. He is engaged in teaching and research since last 20 years. He has more than 80 research papers in journals of international and national repute. He has guided 18 students for their doctorate in computer science. He is also referee of various international and national journals like International Journal of Uncertainty, Fuzziness and Knowledge Based Systems by World scientific publishing cooperation Ltd, International Journal of Engineering, Iran, IEEE Transaction of fuzzy systems and European journal of operation research by Elsevier. He is also the regular member of machine intelligence Research Labs (MIR Labs), scientific network for innovation and research excellence (SNIRE), Auburn, Washington, USE, http://www.mirlabs.org, since 2012.His Google citation indices are 9, i10-index is 8 and he has 257 citations.



Shweta Vishnoi is an Assistant Professor in the Department of Physics at, NIET Greater Noida, India. She received her M.Sc and M.Phil degrees in Physics from Ch. Charan Singh University(CCSU), India. She received her Ph.D degree in 2014 from CCSU, India. She has total teaching experience of fifteen years. Her area of research is Nanotechnology, Modelling and Simulation and data analytics.